10 Grand Challenges

Jordan Pollack
Brandeis University
pollack@cs.brandeis.edu

1) Computational Politics 1: Budgeting and Voting Systems

   So far, computerization has had little impact on politics beyond the collection and automation of mailing lists. **A grand challenge** would be to design and build a scaleable transparent budgeting tool, which can be used by small organizations, like churches and schools, as well as larger ones like cities and states. It would have the form of a dynamic piechart, where each participant could arrange their own interest, and all the participants choices could be combined, uniformly or hierarchically, into a whole budget. Each person could observe--in real time--the effect of their choices on the political process of deciding how to slice the pie.

2) Computational Politics 2: Living History

   While the Internet has let media resell or reuse their writers' contributions, but hasn't changed the centralized dissemination of news and factoids which create history, although there is a thriving conspiracy web where all sorts of misinformation and urban legends can be found. **A grand challenge** would be to employ the same dynamic web content technology used in BLOGs and WIKIs, together with some Bayesian belief network theory, to build a massive network of story nodes which support and refute each other, with an Ebay-like system of rating informants, reporters and historians (who are responsible for sustaining particular interpretations of events, which may oscillate over time.

3) Repairing Email

   Let's face it: Email has been broken by its own success. There is too much email and voicemail and electronic faxes, not to mention spam, viruses, address changes notifications, and confirmations of meetings. Personal computers, webmail boxes, and email clients are not up to the task of managing a lifetime of email content. **A grand challenge** would be to re-envision email so the storage and intelligence could be e managed in the middle of the network, not at the edge.

   Pundits are predicting 300 messages a day, 50% with attachments or HTML by 2004. Spammers can use your own SMTP server to spam you. Current filtering fails because it is too greedy and infringes on free speech. There is a need for free speech and political pamphleteering using anonymity, but we could also have a opt-in community, where every day, a new random key is generated for each participating email server, and spread through a distributed database using secure channels. Each sender on that server would include its key in a header, when talking to other servers in the system, which then could be used by receiving system to validate the sender. Since email is easy to spoof but rarely intercepted, having keys in the header could be s useful for prioritization and filters.

4) Repairing Copyright

   The mechanical production of books and records which lead to copyright as a social and legal convention has been superceded. Individual piracy is rampant on the one hand, yet on

the other, the content industry presumes that a publisher can set any price and print as many copies of whatever as the market will bear, protected by government monopoly! But a real market does not allow price fixing. It does not allow an infinite supply, unless the price is zero. So **a grand challenge** is the free market in intellectual property we don't quite have yet.

With a couple of small changes to the way content (software, books, records, art, video) is licensed, we can have a real system of intellectual property, in which you actually "own" what you buy, and can sell it when you are done, as long as you cannot keep a copy.

1) What are being bought and sold are not the media objects, but permanent use and resale licenses which survive changes in media and information. 2) There is no fee for upgrades, so there is no price difference between new and used licenses; they are determined by a market. 3) The number of such licenses is finite in the way shares in a corporation are limited. 4) A public market with real time trading of rights to use via brokers, market makers, and transfer agents will prevent loss or duplication. See jordanpollack.com/softwaremarket.

5) Repairing Software

A grand challenge is to end software piracy, software unreliability, and software monopoly. Such a free market system where the value of software fluctuates based on quality, and where no "competitive upgrades" can terminate the value of a competitive license, will lead to a value placed on reliability and interoperation, rather than flaws to drive lock in acceptance of the next upgrade. Software piracy will decrease as owners realize they will be depressing the future value of their own equity, rather than just the wealthy publisher. Oligarchies of tools (e.g. Word, Ami, Wordstar, Wordperfect,) will continue to hold market share for converged standard file formats (US rich document standard), rather than a single tool creating a closed and proprietary format to lock in monopoly and force competitors into obsolescence.

6) Repairing Telecommunications

From a birds eye view, the high tech industry is in the doldrums not because of the burst bubble, but because there is a "nitrogen cycle" which has been broken, *Shumpetered* by the onset of the Internet. We don't need as much long distance or faxing since email is cheaper. Local "unlimited" flat rate plans did not anticipate dialup Internet: Users adapt to use more and more of their "free" resource, until the provider is losing money on each customer. The debt load of the Telcos is threatening the whole economy, not just high tech. To outline this cycle in a cartoonish way: People pay utility bills to the telephone company, based on usage. Telco's buy equipment (Boxes) and use a whole economy of other services to maintain and service their network. Box companies buy hardware and software. Software creates jobs with people who pay their utility bills.

Unfortunately, to defeat Compuserve and Prodigy, AOL came up with a Flat Rate unlimited package, creating customer volume. It could, for many years, make up lost revenues from advertising and from equity in IPO's of its partners. All other ISP's, including Telco ISP's are capped by AOL's rate. Users adapt to use more and more of their "free" resource, until the provider is losing money on each customer. Broadband allows 100x the capacity at only 2x the flat rate. The hope for wireless data is to restore "usage" based pricing, as DOCOMO has done in Japan, but because of the fiscal crises, wireless data will remain artificially

overpriced in the US. In Israel, cell calls are 1c/minute. Therefore, **a grand challenge** is to restore the nitrogen cycle of technology by using computing to create a metered way to restore usage based pricing which provides normal profits.

7) Repairing Education

The Internet has not yet been used to revolutionize education, even at the primary level. Entrenched bureaucracies are actually not evil, but a set of human agents and rules for an economic game which has found the equilibrium point. The best way to spend money is not on technology, but to lower the teacher/student ratio. It is **a grand challenge** to devise a technological way to lower this ratio. I think that distributed environments of simulation and gaming, in which multiple participants can safely interact can create a virtual community of learners. We need new networked educational software systems in which students at different grades become *defacto* teachers of other students, even as the 2000 year old Nash Equilibrium of education--one adult in a room with 20 children--lies undisturbed.

8) Increasing Software Complexity

My first programming language was APL, a very high level language. It allowed me to write more code in an afternoon than a team of three Fortran programmers could do in a month. However, APL fell from favor because computer time was valued more than human time, because ASCII terminals rose to prominence, and because self-modifying code and branching were prohibited. Today, we are still programming at the same abstraction level, with the same kind of programming languages. The Fortran Library begat the Turbo Pascal Library begat the Visual C library begat the Java Library.

Yet, Software complexity is stuck where it was 20 years ago, with maximum software being between 10 and 100M lines of code before it collapses under its own weight. Hardware engineers laugh at the failure of software compared to their marvelous achievements. Given a circuit for a byte of memory, a chip with a million bytes is an engineering achievement! However, in software, a million copies of something is just a second line of code, the do loop. In the last 15 years of Moore's law progress, Windows XP boots exactly as fast as Windows 3.1 booted on a 386! **A grand challenge** is to use the extra CPU time to enable more software complexity with less human toil.

9) Precise Simulation

Robotics is held up by the software complexity problem as well. For 50 years, people have built robots by building an electro-mechanical machine, then trying to program the control for it. Doesn't work. Most robots are animatronic puppets, scripted by humans. Robots, outside of industrial automated machine tools, usually fail to achieve a return on the engineering investment. A possible way out is to envision a range of design technologies which use more and more computer time to more automate the process of engineering. We have computer aided design, in which the computer merely acts as a stenographer for the engineer. The next step is "semi-automated" design in which computers can search through constrained spaces of design for optimization, such as making systems lighter, or stronger. **A grand challenge** is practical "fully automated design" systems, software which knows enough about the physical behavior of a set of components, that it can experiment inside virtual worlds with different arrangement of components, and predict the behaviors in reality. Automatic design is held up by the poor quality and slow speed of simulation.

10) A National New Age Manufacturing infrastructure

We have seen the ability to take shapes from the mind of a computer, and turn them into actual physical objects. These rapid prototyping machines work with plastics to create 3D models. Other machines work with metal sintering. As computers become better at design, **a grand challenge** is to have a MOSIS-like project for MEMS and Nanotechnology. Even at greater scales than MEMS, we need to push on the technology for building a New Age Manufactury, local plants which can fabricate multi-material objects comprising non-conductor, conductor, magnetics, electronics, force effectors and sensors, with minimal human intervention. As custom created objects begin to approach the cost of mass-produced objects (in which the fixed design costs need to be amortized) we will see a new industrial revolution of individualized low-cost production which will drive demand for design content and exceed the sum of the industrial, semiconductor, and information revolutions.

Bio

Jordan B. Pollack is a computer science professor at Brandeis University. After an early career in the computer industry, he received the Ph.D in 1987 from University of Illinois, and taught at Ohio State University before moving to Brandeis. He has published many well-cited results across AI, machine learning, neural networks, evolutionary computation, complex systems, and artificial life. His laboratory focuses on the question of self-organization, coevolving robot bodies and brains, and peer-to-peer educational technology (demo.cs.brandeis.edu) and is partially funded by DARPA, ONR, NSF, and DOE. The automatically designed and manufactured Robots made front page news in September 2000. He has been interviewed as “Geek of the Week” on both Slashdot.org and Edge.org, and named one of MIT Technology Review’s "TR 10" in January 2001. He has a long-term interest in information property, and advises several start-ups companies, and founded Thinmail, inc, which markets a lightweight multimedia communications service platform.